07 | 故障处理：一切以恢复业务为最高优先级

2020-04-01 赵成

SRE实战手册

进入课程 >

讲述：赵成

时长 15:55　大小 14.59M

你好，我是赵成，欢迎回来。

上一讲我们讨论了 MTTR 的第一个环节，也就是 MTTI（故障发现）阶段的应对措施。今天我们继续来看 MTTR 的另外三个环节，也就是 MTTK、MTTF 和 MTTV 阶段，这三个对应的就是故障诊断、修复与确认，我们一起来看在这些环节应该做好哪些事情。

今天的内容，我会围绕一条基本原则展开，那就是：**在故障处理过程中采取的所有手段和行动，一切以恢复业务为最高优先级**。这也是我们故障处理过程中的唯一目标，没有第二

相信你一定理解这句话我想要表达的意思，但是在执行过程中，我们往往就是很难执行到位。原因是什么呢？

讲到这里，我先分享一个我自己团队的案例，因为这个案例非常典型，其中暴露出来的问题，你很有可能也遇到过。

我们先一起来看下。

一次应急操作导致的故障蔓延

在 2016 年的双十一前，我们蘑菇街策划了一场预热活动，各种优惠券非常诱人，当时确实吸引了大量的用户来参与。这个活动是秒杀类型，瞬时的并发量非常大，我们很乐观地以为做好了十足的准备。

然而真实情况是，一到时间点就出现了用户下单失败问题。当时，我们赶紧采取限流措施，但是发现下单请求仍然大量阻塞，用户访问超时失败。后来进一步排查，发现请求都阻塞在了价格计算的 SQL 语句上。这个信息很快传递到了 DBA 这里，DBA 就马上采取措施：既然有阻塞，那扩容就是最快的手段，先尝试一下再说，所以马上做了 Slave 节点扩容。

因为当时情况紧急，DBA 承受的精神压力也特别大，遗漏了做数据完整性和一致性校验，在数据没有完全同步完成的情况下，就把扩容节点上线了，结果就是阻塞问题有一定缓解，却出现了计费价格错误这种严重的业务故障，也就是优惠券的价格优惠没有用到，用户多付了钱。毫无疑问，紧跟着我们就收到了大量的用户投诉。

继续定位根源问题，我们发现用户的访问模型跟原来预估的不一致，当前的代码逻辑无论如何也支撑不住这样的访问量，唯一的方式就是紧急修改代码，然后做测试验证再上线。但是这样要花费非常长的时间，完全超出了活动时间，所以我们最终不得不临时取消预热活动，通过给用户退费或补发优惠券的方式来安抚客户。

后来，我们复盘总结，认为这次故障的发生和持续蔓延有三个原因。

第一个，故障隔离手段缺失。

之所以引发这样的故障，其实就是没有提前考虑到对应的故障隔离手段，比如有效的限流、降级和熔断。如果这些措施不具备，从技术层面其实已经很难采取有效的手段了，所以最后的结果就是，要么提前终止活动，要么等代码优化完成上线。

因此，这里业务恢复慢，根本原因不是响应不及时，处理流程不高效等等，这个跟处理过程关系不大，而是**从根本上就不具备业务恢复的技术手段**。所以结合你自身的工作情况，也要进行一个反思：是不是业务模型评估得不够准确？在 Design-For-Failure 的高可用设计上做得不够？等等。

第二，关键角色和流程机制缺失。

在处理故障过程中，DBA 发现 SQL 语句执行阻塞了，觉得扩容有效，就马上去做了，并没有一个共同决策和反馈的机制。抛开态度，单从结果上讲就是忙中添乱，导致出现更为严重的衍生故障。

所以，我的总结就是，没有有效的故障应急处理流程，分工不明确，关键角色缺失，没有决策、反馈和通报机制，导致整个过程混乱失控。

第三，没有演练。

处理故障失败，导致预热活动以失败告终，其实也是因为我们缺乏演练，在面对真正的问题时，场面混乱，没有章法。

这里我说的是缺少演练，其实在我实际了解的很多案例中，但凡故障处理不好的，基本就是没有任何演练，这里又分为两种。

一种是我们前面讲的第一个原因，故障隔离手段缺失。要么缺失手段，不知从何入手，要么就是有限流降级等故障隔离策略，但是不敢尝试和演练。为什么呢？就是担心演练有可能也会影响正常业务，但这也侧面说明我们的技术方案仍然不够成熟和完善。因为如果在相对比较宽松的氛围下，都不敢演练不敢操作，那真正出问题的时候就更不敢乱动，或者动了系统会导致更多意想不到的状况。所以，我们应该借助演练的手段，来倒逼我们考虑得更完善才可以。

另一种就是整个应急流程基本跑不起来，要么是人凑不齐，要么是很多团队不愿意配合，只要是这种理由导致的无法演练，基本出现故障后，整个应急状态就是失控的。因为大家都是慌乱状态，平时都缺少配合和协作，在这种高压状态下，大家就更顾不上或者不知道该怎么配合了。

这么复盘下来，故障响应其实是两个方面，技术方面和流程机制方面。技术问题我们这里就不再做详细介绍了，你可以有针对性地看下我第一季课程里的第 21 讲《极端业务场景下，我们应该如何做好稳定性保障？》内容。

这里，我们重点来讨论如何建立有效的故障应急响应机制。

如何建立有效的故障应急响应机制

关于这一点，我们接着上一讲的内容说，当一个问题被定性为故障，这时我们就要成立 War Room，如果是在办公时间，大家可以聚集到同一个会议室，或者同一块办公区域集中处理；如果是非办公时间，可以是视频、电话或微信会议的方式，形成虚拟的 War Room。但无论是真实还是虚拟的 War Room，根本目的是快速同步信息，高效协作。

在特别时期，比如双十一这样的大促活动，我们团队的做法一般是空出一个楼层，让所有参与保障的同事坐在一个楼层办公。在已经形成双十一文化的阿里，他们把这样的办公地点称为“光明顶”，各大高手齐聚于此，共同保障业务和系统稳定。

仅仅有 War Room 这样一个聚集形式还是不够的，既然我们把解决故障类比成战争，我们就一定要有一套相对应的指挥体系才可以，这个体系里面，我认为最重要的就是“关键角色分工”和“沟通机制”。

这里我先给你介绍下 Google 的指挥体系，然后再结合我自己的实践来展开。

关键角色分工

Google 的故障指挥体系，是参考了美国消防员在处理 1968 年森林大火时建立的应急指挥体系，体系中有三个核心角色。

Incident Commander，故障指挥官，简称为 IC。这个角色是整个指挥体系的核心，他最重要的职责是组织和协调，而不是执行，下面所有的角色都要接受他的指令并严格执行。

Communication Lead，沟通引导，简称 CL。负责对内和对外的信息收集及通报，这个角色一般相对固定，由技术支持、QA 或者是某个 SRE 来承担都可以，但是要求沟通表达能力要比较好。

Operations Lead，运维指挥，简称 OL。负责指挥或指导各种故障预案的执行和业务恢复。

这里其实还有一类角色，虽然不在指挥体系内，但是也非常重要，我们也要做一下介绍：**Incident Responders，简称 IR。**就是所有需要参与到故障处理中的各类人员，真正的故障定位和业务恢复都是他们来完成的，比如具体执行的 SRE、网络和系统运维、业务开发、平台开发、网络或系统运维、DBA，甚至是 QA 等。

流程机制

在我自己团队的具体实践和场景中，我们按过程来分，会有如下的一个流程机制。

1. 故障发现后，On-Call 的 SRE 或运维，最一开始就是 IC，有权召集相应的业务开发或其它必要资源，快速组织 War Room。
2. 如果问题和恢复过程非常明确，IC 仍然是 SRE，就不做转移，由他来指挥每个人要做的具体事情，以优先恢复业务优先。
3. 如果问题疑难，影响范围很大，这时 SRE 可以要求更高级别的主管介入，比如 SRE 主管或总监等，一般的原则是谁的业务受影响最大，谁来牵头组织。这时 SRE 要将 IC 的职责转移给更高级别的主管，如果是全站范围的影响，必要时技术 VP 或 CTO 也可以承担 IC 职责，或者授权给某位总监承担。

从我们的实践经验来看，如果是大范围的故障，一般就是平台技术总监或电商业务的技术总监来承担 IC 职责，接下来他就可以从更高的层面组织和协调资源投入并有效协作。这时 SRE 会回归到 OL 的职责上，负责组织和协调具体的执行恢复操作的动作。

反馈机制

有了角色分工，也明确了流程，那整个故障响应是否高效的关键就是沟通机制了，这里我重点分享下我们团队在反馈上的一些经验和心得。

反馈的重要性是再怎么强调都不为过的。有时涉及的团队和人员比较多，很多具体执行的同事就只顾闷头定位和排查了，往往没有任何反馈，甚至做了很多的操作也是自行决定，不做通报。这时就会出现上面案例说的场景，极有可能衍生故障或者导致故障更大范围的蔓延，这是极为影响协作效率和决策的。

我们一般要求以团队为单位，每隔 10～15 分钟做一次反馈，反馈当前处理进展以及下一步 Action，如果中途有需要马上执行什么操作，也要事先通报，并且要求通报的内容包括对业务和系统的影响是什么，最后由 IC 决策后再执行，避免忙中出错。

这里我要强调一点，**没有进展也是进展，也要及时反馈**。很多团队和成员往往会抱怨，我专心定位问题还没结果呢，有什么好反馈的呢？但是没有进展也是进展，IC 会跟 OL 以及团队主管决策是否要采取更有效果的措施，比如 10 分钟之内还没定位结果，可能我们就会选择做有损的降级服务，不能让故障持续蔓延，这个时候，反馈就显得尤为重要。

同时，在这个过程中，为了尽量减少对执行者的干扰，让执行者能够更聚焦，我们一般要求**团队 Leader 来收集反馈并传递 IC 的指令**。CL 也要收集信息，他要做的不是传达指令，而是在更大范围内同步汇总后的信息，同时还要整理信息传递给外部联系人。

到这里，我们会发现在故障处理过程中，特别是影响范围较大的故障，其实还会涉及一类非技术群体，这一类角色 Google SRE 并没有提，但是我们实践中会发现这一部分人也非常关键，比如**客服、PR 以及商家和其它各类合作代表**等等。

为什么说他们也非常关键呢？因为我们要知道，故障之所以带来巨大的压力，很大程度上是因为业务影响导致的用户投诉和抱怨，以及合作伙伴们利益受损带来的各类负面情绪，这类抱怨和情绪有时候还会直接发泄到公司高管那里，这时压力就会层层透传下来。

所以，除了要做好怎么**快速恢复业务，信息同步的及时和透明也非常关键**，并且有些安抚措施必须要快速执行到位。

比如，现在更多的用户习惯在朋友圈、微博或知乎这样的社交平台发表意见，所以在用户不了解背景信息的时候，会表达一些情绪，甚至有些媒体为吸引眼球做夸大或抹黑的报道。为了遏制这些负面影响，公司 PR 要出面与平台或媒体沟通，并通过这些社交平台的官方账号做统一的信息同步，做到及时透明。

再比如，故障可能会对用户、商家或合作渠道造成利益损失，这时就需要运营和产品团队及时制定补偿策略，例如补偿优惠券等，并通过客服同步给用户和合作伙伴，这些对于安抚用户情绪至关重要。

所以，上述 CL 的信息收集、汇总以及与非技术团队的沟通就至关重要，怎么沟通呢？一定不是用技术术语，而是**以尽量业务化的语言描述，并且要给到对方大致的预期**，比如我们正在做什么，大致多长时间会恢复，如果不能恢复，大约多长时间内会给一个反馈等等。

这些措施的有效执行，会大大减少故障之外的干扰因素，让技术团队能够更专注于故障本身的处理。相反，如果这些事情不重视，我们就会发现，会有各种渠道和各种层级的领导以及接口人来质问你：“到底什么时候可以恢复？你们的系统为什么总是这么不稳定？”

所以，如果故障的影响范围很广，那我们就要考虑得尽量周全，这时的**故障处理在一定程度上，就不单单是技术团队的问题了，而是需要整个公司都投入资源的**。

这样需要全公司共同配合的事情，想要很顺畅地执行，那就一定要在日常做演练。同时有些事项比如反馈信息的模板，对外通报的模板都要提前建立好，每个角色只要按照模板填写内容即可，可以极大地保证沟通效率。

总结

好了，我们来做下总结。

故障处理过程中效率如何，其实取决于三个因素：技术层面的故障隔离手段是否完备；故障处理过程中的指挥体系是否完善，角色分工是否明确；故障处理机制保障是否经过足够的演练。

我们可以看到，想要高效地处理故障，并不是说我在技术层面做到完备就可以了，这是一个需要体系化建设和反复磨炼才能最终达成的目标。

思考题

最后，我留一个问题给你，想听听你的看法。

我在本节课中提到，故障应急过程中要有一个指挥官角色，他有权调动所有必要的角色参与到应急过程中，那么在你的公司或团队中，是由谁来指挥，或者说是由谁来承担这个角色的呢？

欢迎你在留言区分享，也希望你把本篇文章转发给你身边的朋友。

我是赵成，我们下节课见。

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇　06 | 故障发现：如何建设On-Call机制?

精选留言 (1)

写留言

leslie
2020-04-01

这个问题应当是不同公司不同情况吧：有些企业或是技术总监，有些会是部门主管；这其实涉及到的最关键问题就是权限-足够的沟通和协调权限，这种可能涉及跨部门操作的事情只能是公司的中高层。
甚至当直接协调人不在时会有相关业务部门的同级别人员去承担相应的事情。

展开

作者回复: 不同企业情况不同，这里最重要的是要有足够的授权给到。